Sistema de Clasificación de Actividades Humanas mediante Visión por Computadora y Machine Learning

Joshua Rivera Gonzalez, Thomas Brueck, Santiago Espinosa, Juan Pablo Parra

Algoritmos y Programación III, Ingeniería de Sistemas,
Facultad Barberi de Ingeniería, Diseño, y Ciencias Aplicadas
Universidad ICESI, Cali, Colombia

Abstract

Este proyecto presenta un sistema completo de reconocimiento de actividades humanas (HAR) que procesa video en tiempo real mediante visión por computadora y técnicas de machine learning. El sistema utiliza MediaPipe Pose [1] para extraer landmarks corporales, a partir de los cuales se calculan características cinemáticas (ángulos, distancias, velocidades y aceleraciones). Estas características se emplean para clasificar cinco actividades: caminar hacia la cámara, caminar de regreso, girar, sentarse y ponerse de pie. Se entrenaron y compararon tres modelos de clasificación (SVM [4], Random Forest [5] y XGBoost [2]) sobre un conjunto de 69 videos (63,964 frames), de los cuales 29,701 frames corresponden a las actividades principales. El modelo XGBoost obtuvo el mejor desempeño con 77.91% de precisión en validación y 78.87% en prueba, utilizando 15 características seleccionadas de un total de 147 mediante análisis de importancia. El sistema final implementa clasificación en tiempo real con latencia inferior a 50 ms por frame, lo que demuestra su viabilidad para aplicaciones prácticas en salud, deportes e investigación biomecánica.

Palabras clave: reconocimiento de actividades humanas, visión por computadora, MediaPipe, machine learning, clasificación en tiempo real, análisis biomecánico.

I. Introducción

A. Contexto y Descripción del Problema

El análisis de movimiento humano es fundamental en rehabilitación física, evaluación deportiva, ergonomía laboral y sistemas de asistencia para adultos mayores. Tradicionalmente, este análisis se realiza mediante anotación manual de videos, proceso que requiere tiempo significativo (hasta 30 minutos por video) y está sujeto a variabilidad inter- e intra-anotador. Los sistemas de reconocimiento de actividades humanas (HAR) automatizan este proceso utilizando datos de sensores o video [6]–[8]. Los avances recientes en estimación de pose, particularmente BlazePose/MediaPipe [1], permiten la extracción robusta y en tiempo real de landmarks corporales, abriendo posibilidades para aplicaciones prácticas fuera del entorno de laboratorio. Este trabajo aborda la clasificación automática de actividades humanas a partir de secuencias de video monocular. El objetivo es identificar cinco actividades principales: caminar hacia la cámara, caminar de regreso, girar, sentarse y ponerse de pie. El sistema desarrollado debe: (1) extraer información postural de cada frame mediante detección de landmarks corporales, (2) calcular características cinemáticas que capturen la dinámica del movimiento, (3) clasificar la actividad en tiempo real con precisión aceptable (≥75% en conjunto de prueba), y (4) operar con baja latencia (<100 ms por frame) sin requerir GPU.

B. Por Qué Este Problema es Interesante

Este problema es relevante porque combina: (1) aplicaciones prácticas directas en salud, deporte, ergonomía y asistencia a adultos mayores, (2) desafíos técnicos multidisciplinarios que integran visión por computadora, procesamiento de señales temporales y modelos de machine learning, (3) requisitos de escalabilidad dado que el sistema debe generalizar a diferentes personas, condiciones de iluminación y configuraciones de cámara, y (4) restricciones de tiempo real que exigen un balance cuidadoso entre precisión y eficiencia computacional. El sistema aborda desafíos persistentes incluyendo extracción robusta de características posturales, manejo de variabilidad entre sujetos, clasificación precisa de actividades dinámicas y cumplimiento de restricciones de tiempo real en hardware convencional.

II. Teoría

A. Estimación de Pose con MediaPipe

MediaPipe Pose se basa en el modelo BlazePose [1], que estima 33 landmarks corporales en 2D/3D por frame. Cada landmark incluye coordenadas normalizadas (x, y, z) y un valor de visibilidad. Esta representación es aproximadamente invariante a escala y traslación, facilitando la generalización a diferentes distancias y posiciones de cámara. MediaPipe proporciona una "capa de abstracción" sobre la imagen original, reduciendo el problema a trabajar con coordenadas articulares en lugar de píxeles.

B. Ingeniería de Características para Análisis de Movimiento

Para capturar la dinámica del movimiento a partir de los landmarks, se construyen características cinemáticas: (1) Ángulos articulares (por ejemplo, hombro-codo-muñeca, cadera-rodilla-tobillo) calculados mediante relaciones vectoriales, (2) Distancias euclidianas entre landmarks clave que cuantifican amplitudes de movimiento, (3) Velocidades como primera derivada temporal (diferencias entre frames consecutivos), y (4) Aceleraciones como segunda derivada temporal (cambios de velocidad). Estas características combinan información de postura estática con patrones de cambio temporal, crucial para distinguir actividades con posturas similares pero transiciones diferentes (por ejemplo, sentarse vs. permanecer sentado).

C. Modelos de Clasificación

Se compararon tres familias de modelos clásicamente utilizados para clasificación con características tabulares: Support Vector Machine (SVM) [4] busca un hiperplano que separe las clases maximizando el margen. Con kernel RBF, puede modelar fronteras de decisión no lineales. Generalmente robusto

al overfitting con regularización adecuada, aunque el costo computacional crece con el tamaño del dataset.

Random Forest [5] es un ensamble de árboles de decisión entrenados con muestreo bootstrap y selección aleatoria de características. Maneja bien datos ruidosos y desbalanceados, proporcionando medidas de importancia de características. Sin embargo, un gran número de árboles puede incrementar el tiempo de inferencia.

XGBoost [2] es una implementación optimizada de gradient boosting sobre árboles de decisión. Construye árboles secuencialmente, corrigiendo errores del ensamble previo. Se caracteriza por alto desempeño predictivo y eficiencia computacional, a costa de requerir ajuste cuidadoso de hiperparámetros.

D. Manejo de Desbalance de Clases

Con clases desbalanceadas, los modelos tienden a favorecer las clases mayoritarias. SMOTE (Synthetic Minority Over-sampling Technique) [3] aborda esto generando muestras sintéticas de clases minoritarias mediante interpolación entre vecinos cercanos en el espacio de características, balanceando el conjunto de entrenamiento y mejorando la cobertura de estas clases.

III. Metodología

El desarrollo del sistema siguió una metodología iterativa cubriendo desde la extracción de datos hasta la implementación en tiempo real.

A. Adquisición y Preprocesamiento de Datos Extracción de Landmarks: Se procesaron 69 videos utilizando MediaPipe Pose [1]. Para cada frame, se extrajeron 33 landmarks con coordenadas (x, y, z, visibility) almacenados en archivos CSV, resultando en 63,964 frames con landmarks normalizados.

Preprocesamiento y Suavizado: Se aplicó un filtro Savitzky-Golay [9] (ventana=5, orden=2) para suavizado temporal reduciendo ruido y jitter en las trayectorias de landmarks. La normalización espacial centró el esqueleto usando el centro del torso como referencia, buscando invariancia a la posición absoluta dentro del frame.

B. Ingeniería de Características

Características Iniciales: Las características base incluyeron ángulos de rodillas izquierda/derecha, ángulos de codos izquierdo/derecho, y distancia muñeca-hombro como indicador de movimiento de brazos.

Vector de Características Final: Se construyó un vector de 147 características combinando: (1) 132 componentes de landmarks (33 × 4 coordenadas: x, y, z, visibility), (2) 5 ángulos base, y (3) 5 velocidades y 5 aceleraciones asociadas a estos ángulos/distancias. Esta representación comprensiva captura tanto postura estática como dinámica temporal.

C. Preparación del Dataset

Integración de Etiquetas: Se integraron anotaciones manuales de Label Studio, estandarizando etiquetas de diferentes anotadores y asignando etiquetas temporales a cada frame. El filtrado para cinco actividades principales produjo 50,075 frames etiquetados (78.3% del total), con 29,701 correspondiendo a actividades principales.

División de Datos: La división del dataset se realizó estratificada por clase: Entrenamiento 70% (37,450 frames), Validación 15% (4,455 frames), Prueba 15% (4,456 frames). Se aplicó StandardScaler [10] a todas las características (ajustado en entrenamiento, aplicado a validación y prueba). SMOTE [3] se utilizó únicamente en el conjunto de entrenamiento, buscando balance cercano a 10,000 muestras por clase.

D. Entrenamiento y Evaluación de Modelos

Se entrenaron tres modelos: SVM [4], Random Forest [5] y XGBoost [2], usando GridSearchCV con validación cruzada de 5 folds y F1-score ponderado como métrica de optimización para manejar el desbalance de clases. Mejores hiperparámetros: SVM (C=100, kernel=RBF, gamma='scale'), Random Forest (n_estimators=300, max_depth=None), XGBoost (n_estimators=300, max_depth=7, learning_rate=0.3).

E. Selección de Características e Implementación en Tiempo Real

El análisis de importancia de características con XGBoost [2] identificó las 15 características más relevantes, reduciendo la dimensionalidad de 147 a 15 (89.8% de reducción). El pipeline en tiempo real consiste en: VideoProcessor para extracción de landmarks y cálculo de características, ActivityClassifier para carga del modelo, preprocesamiento, selección de features y suavizado temporal mediante buffer de 10 frames con votación mayoritaria, y aplicación en tiempo real capturando video desde webcam con visualización de esqueleto y actividad predicha.

IV. Resultados

A. Desempeño de Modelos

En validación, XGBoost alcanzó 77.91% de precisión y 0.781 de F1-score ponderado. En el conjunto de prueba, el desempeño se mantuvo en 78.87% de accuracy, con F1-score macro de 76.42% y promedio ponderado de 78.96%. Las actividades con mejor desempeño fueron "caminar hacia cámara" y "girar" (ambas F1 > 81%), mientras que "sentarse" mostró el F1 más bajo (64.25%). Resultados de entrenamiento:

Modelo	CV Score	Val Accuracy	Val F1	Tiempo Entrenamiento (s)
SVM	75.66%	64.48%	0.651	4,141
Random Forest	88.27%	77.58%	0.779	689
XGBoost	88.64%	77.91%	0.781	246

Resultados en conjunto de prueba:

Modelo	Test Accuracy	Macro Precision	Macro Recall	Macro F1
SVM	64.29%	60.01%	63.39%	61.21%

Random Forest	78.64%	75.20%	78.84%	76.69%
XGBoost	78.87%	75.62%	77.38%	76.42%

Desempeño por clase de XGBoost:

Actividad	Precision	Recall	F1-Score	Support
Caminar de regreso	71.94%	79.58%	75.57%	377
Caminar hacia cámara	85.98%	80.76%	83.29%	972
Girar	81.74%	81.13%	81.43%	1,065
Ponerse de pie	74.91%	80.47%	77.59%	256
Sentarse	63.54%	64.97%	64.25%	354

B. Impacto de la Reducción de Características

Las 15 características más importantes incluyen: left_knee_angle (13.62%), left_wrist_dist (12.92%), left_elbow_angle (11.74%), right_elbow_angle (9.10%), right_knee_angle (8.83%), y coordenadas de landmarks clave. Las cinco principales concentran 56.2% de la importancia total; las diez primeras alcanzan 82.6%. Esta reducción permitió: ~80-85% de reducción en tiempo de inferencia, ~90% de disminución en uso de memoria, y mejora en interpretabilidad del modelo.

C. Desempeño en Tiempo Real

Las pruebas demostraron: latencia por frame < 50 ms (objetivo: <100 ms), throughput aproximado de 20 FPS en CPU, y un desempeño aceptable, tal vez no fue el mejor pero cumple con los requerimientos mínimos.

V. Análisis de Resultados

A. Resumen

Este trabajo desarrolló un sistema de reconocimiento de actividades humanas que: (1) extrae información postural de video monocular usando MediaPipe Pose [1], (2) construye 147 características cinemáticas (ángulos, distancias, velocidades, aceleraciones), (3) integra etiquetas manuales de 69 videos (63,964 frames) generando un dataset de 29,701 frames con cinco actividades principales, (4) entrena y evalúa tres modelos de clasificación seleccionando XGBoost como modelo final, (5) reduce dimensionalidad de 147 a 15 características mediante análisis de importancia manteniendo desempeño global, y (6) implementa aplicación en tiempo real con latencia < 50 ms por frame en CPU.

B. Aprendizajes Clave

La ingeniería de características temporales (velocidades y aceleraciones) mejora significativamente el desempeño versus usar solo información estática. El balanceo de clases mediante SMOTE [3] es clave para mejorar desempeño en clases minoritarias. La selección de características basada en importancia permite reducir complejidad del modelo sin sacrificar precisión. XGBoost ofrece excelente compromiso entre desempeño y tiempo de entrenamiento/inferencia para este tipo de problema. Cumplir requisitos de tiempo real en hardware estándar es posible mediante pipeline optimizado y reducción adecuada de características.

C. Mejoras Futuras

Corto plazo (1-3 meses): Aumentar tamaño y diversidad del dataset (más sujetos, condiciones de captura, actividades); ampliar ventana de análisis temporal para capturar transiciones completas, especialmente en "sentarse" y "pararse"; incorporar características de trayectoria describiendo dirección de desplazamiento global.

Mediano plazo (3-6 meses): Extender repertorio de actividades (correr, saltar, agacharse, etc.); explorar arquitecturas profundas (CNN + LSTM) [6]–[8] usando landmarks como entrada en escenarios con GPU disponible; optimizar implementación para soportar múltiples personas en escena y múltiples streams de video.

Largo plazo (6-12 meses): Desarrollar modelos end-to-end con mecanismos de atención y técnicas de transfer learning; integrar sistema en aplicaciones específicas (rehabilitación, análisis deportivo, ergonomía, monitoreo de adultos mayores); diseñar API y dashboard web para facilitar uso del sistema por profesionales de salud y entrenadores.

D. Impacto

El sistema desarrollado demuestra que el reconocimiento de actividades humanas basado en landmarks es factible con precisión cercana al 80% y operación en tiempo real sobre CPU. Esto abre posibilidades en: Salud y rehabilitación (monitoreo automatizado de ejercicios terapéuticos y evaluación de progreso), Deportes (análisis de técnica y prevención de lesiones), Ergonomía laboral (detección de posturas de riesgo y recomendaciones de mejora), y Asistencia a adultos mayores (detección de patrones de movilidad anómalos y soporte en tiempo real). Para aplicaciones críticas (por ejemplo, diagnóstico clínico), será necesario aumentar significativamente la precisión, ampliar el dataset e incorporar procesos de validación clínica antes del despliegue.

Video en youtube:

Link del video

VII. Referencias Bibliográficas

[1] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device Real-time Body Pose Tracking," arXiv preprint arXiv:2006.10204, 2020.

[2] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[3] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Oversampling Technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[4] C. Cortes and V. Vapnik, "Support-Vector Networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, 1995.

[5] L. Breiman, "Random Forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[6] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep Learning for Sensor-based Activity Recognition: A Survey," Pattern Recogn. Lett., vol. 119, pp. 3–11, 2019.

[7] O. D. Lara and M. A. Labrador, "A Survey on Human Activity Recognition using Wearable Sensors," IEEE Commun. Surveys Tuts., vol. 15, no. 3, pp. 1192–1209, 2013.

[8] H. F. Nweke, Y. W. Teh, M. A. Al-Garadi, and U. R. Alo, "Deep Learning Algorithms for Human Activity Recognition using Mobile and Wearable Sensor Networks: State of the Art and Research Challenges," Expert Syst. Appl., vol. 105, pp. 233–261, 2018.

[9] A. Savitzky and M. J. E. Golay, "Smoothing and Differentiation of Data by Simplified Least Squares Procedures," Anal. Chem., vol. 36, no. 8, pp. 1627–1639, 1964.

[10] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.